УДК 004.891

РАЗРАБОТКА ИНСТРУМЕНТАЛЬНОГО СРЕДСТВА ПОСТРОЕНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ ОБЪЕКТНО-ОРИЕНТИРОВАННЫХ МОДЕЛЕЙ ДЛЯ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ

Г.В. Стародубцев, М.П. Силич*, В.А. Силич

Томский политехнический университет
E-mail: glebst@mail.ru
*Томский государственный университет систем управление и радиоэлектроники
E-mail: smp@muma.tusur.ru

Предложена архитектура и реализация информационной системы для поддержки принятия решений на основе объектно-ориентированных моделей и функциональных зависимостей атрибутов. Показана возможность интеграции в модель различных методов искусственного интеллекта.

Введение

Эволюция классических систем искусственного интеллекта (статические экспертные системы, нейронные сети, экспертные системы реального времени), переход к их массовому использованию при решении практических задач, во многом опирается на объединение с технологиями традиционного программирования, определяющими рынок информационных технологий. Большая часть рынка интеллектуальных систем приходится на долю систем поддержки принятия решений, создание которых требует решений из области прикладного программирования, например, сбор и хранение данных от внешних источников, параллельные вычисления, организация пользовательских интерфейсов, защита информации. Необходимость выполнения этих требований, а также интеграции с существующими и действующими коммерческими системами, ведет к вытеснению инструментальных средств, основанных на классических языках искусственного интеллекта (LISP, Prolog), в пользу более гибких оболочек (например, интегрированная среда G2 от Gensym, RT Works) [1].

Актуальными чертами современных инструментальных средств искусственного интеллекта становятся:

1. Комбинация различных методов искусственного интеллекта.
2. Интеграционный потенциал. Возможность легкой интеграции с другими информационными технологиями и системами (СУБД, CASE, контроллеры, хранилища данных и пр.).
3. Масштабируемость. Поддержка распределенной архитектуры повышает надежность и общую производительность.
4. Переносимость, независимость от вычислительной платформы.
5. Следование стандартам и открытость интерфейсов.
6. Проблемная ориентация.

Одним из классов проблем, решаемых экспертными системами, является поддержка принятия проектных решений (системы проектирования).

На ранних стадиях разработки проектов, особенно инновационных, трудно оценить будущие затраты и качественные параметры реализации. Объективные знания рассредоточены между членами команды, принятие решений происходит под действием многих неопределенностей. Неопределенность – а значит и риск – возникает по многим причинам, включая еще не принятые проектные решения, существование альтернативных моделей поведения, нечеткие внешние факторы, недостаток спецификаций, неполнота информации, использование приближений при вычислении параметров проекта.

Существующие инструменты для интеллектуальной поддержки комплексной и нечеткой информации на различных этапах развития проекта, в значительной мере специализированы по промышленным областям и труднодоступны для инновационных проектов. Например, среда RiTo для проектов в автомобильной промышленности, программный комплексы Merak (Шлюмберже) и ARIES (Landmark Graphics) для нефтегазовой отрасли, специализированные решения на основе платформы G2 (Gensym) [2–5].

В данной работе рассматривается разработка комплекса для моделирования и интеллектуальной поддержки принятия проектных решений в более широком спектре приложений. Изначально он планировался как фундамент для интеграции различных интеллектуальных технологий и средство быстрого построения моделей проектов с помощью графических объектов и библиотек знаний.

Предлагаемая технология близка к идеям из работ по управлению разработкой продукта (*product design*) [2, 6, 7], но в сочетании представления объектной структуры проекта с нечеткими атрибутами, формализованными связями между элементами, возможностью подключения различных взаимодополняющих интеллектуальных методов (декларативные правила, нечеткие множества, функциональные зависимости, нейросети и др.), дает в итоге возможность моделирования проектов в различных областях с учетом неопределенности.

Объектно-ориентированная модель

Теоретической основой для данной разработки является объектно-ориентированный подход к описанию сложных систем [8, 9].

Формально, объектная модель проекта может быть представлена в виде набора дочерних моделей (рис. 1):

$$M =< M^S, M^C, M^A, M^O >,$$

где M^S – модель компонент (подсистем) системы, M^S – классов, M^A – зависимостей атрибутов, M^O – объектов (экземпляров данных).

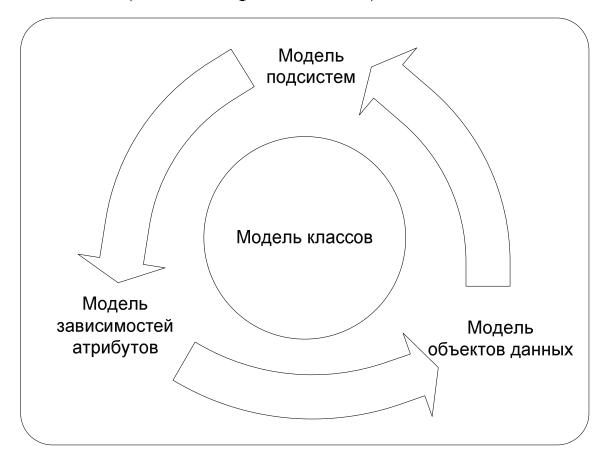


Рис 1. Элементы объектно-ориентированной модели

В соответствии с подходом, предметная область представляется в виде совокупности взаимосвязанных компонент-подсистем, описываемых набором характеристик и выполняемых ими действий. Каждой подсистеме сопоставляется класс, определяющий структуру ее описания (состав атрибутов и методов). На множестве атрибутов классов устанавливаются отношения зависимости в виде аналитических выражений или правил продукции. Далее на базе каждого класса формируется один или несколько экземпляров (объектов) подсистемы, содержащих конкретные значения атрибутов, дополненные коэффициентами уверенности.

Для более удобного отражения множества состояний или вариантов подсистемы используется мультиобъект – набор экземпляров, выделенных в соответствии с некоторым признаком, в качестве которого выступает заданный ключевой атрибут или комбинация нескольких ключевых атрибутов. Каждому значению признака соответствует свой объект-экземпляр.

Отношения зависимости атрибутов описываются на основе абстрактного класса «Зависимость», содержащего список атрибутов-аргументов и закономерность (выражение, описывающее, как определяется значение атрибута-функции на основе аргументов). Модель зависимостей используется для определения значений целевых атрибутов на основании исходных.

Основные компоненты инструментального средства

Информационная система моделирования проектной деятельности во многом схожа по архитектуре с классической структурой экспертных систем. Основные компоненты приведены ниже (рис. 2).

База знаний состоит из локальных библиотек данных, содержащих определения всех объектов конкретной модели, а также общей библиотеки, в которой хранятся все «встроенные» в систему описания объектов. Локальные библиотеки создаются пользователем в процессе моделирования и содержат:

- структуры данных (подсистемы, классы, атрибуты, зависимости между атрибутами, экземпляры объектов);
- управляющие данные (правила, формулы в зависимостях между атрибутами);
- исходные данные, введенные пользователем;
- вычисляемые переменные, которые имеет смысл хранить между запусками модели, и отражающие ее состояние на момент сохранения.

Общая библиотека содержит наборы классов, а также шаблоны структур подсистем для различных предметных областей. Шаблон представляет собой

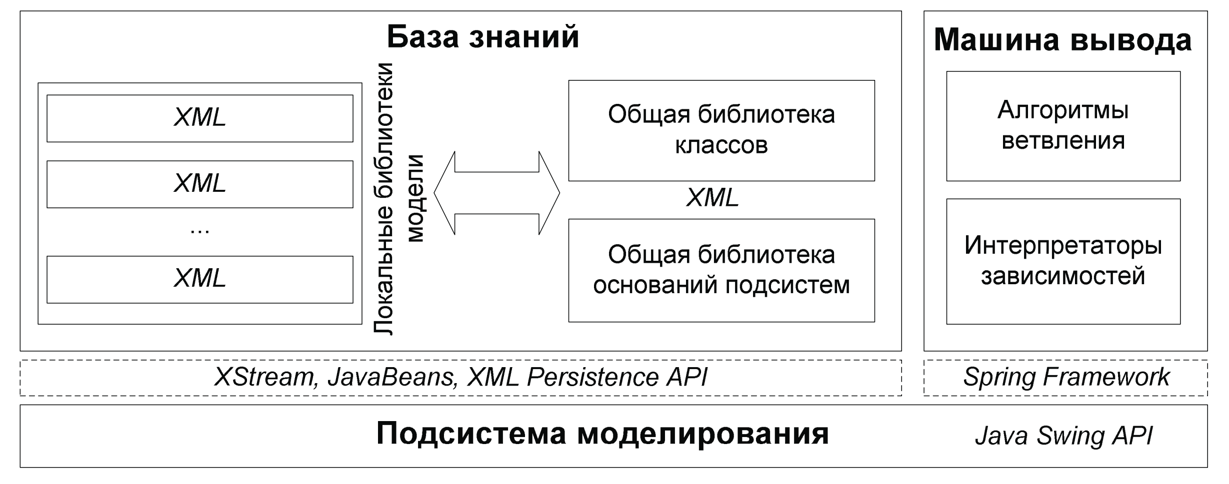


Рис. 2. Архитектура информационной системы

группу подсистем связанных отношением декомпозиции по определенному основанию, например «Жизненный цикл проекта» описывает подсистему «Проект» и связанные с ним дочерние подсистемы «Предварительная оценка», «Разработка», «Реализация», «Внедрение». Доступные наборы классов и шаблонов подсистем отражаются в панели ресурсов графического интерфейса и выбираются индивидуально для каждой модели. Во время работы с моделью, при добавлении новой подсистемы на диаграмму, пользователь может выбрать подходящее основание декомпозиции, и система автоматически добавит дочерние элементы с необходимыми связями.

Предусмотрена возможность экспорта классов и подсистем из локальных библиотек в общую, для дальнейшего повторного использования. Такой подход делает систему более гибкой для использования в различных областях, обеспечивает накопление экспертных знаний и ускоряет процесс построения модели. Технически, база знаний реализована в виде XML-документов с определенной структурой.

Подсистема вывода отвечает за интерпретацию зависимостей между атрибутами объектов моделирования, тем самым, обеспечивая вычисление результатов на массиве введенных исходных данных. В основе машины вывода лежат следующие элементы:

- Набор алгоритмов, обеспечивающих запуск и ветвление вычислений на узлах зависимостей – в настоящее время поддерживаются модифицированный прямой и обратный вывод [10].
- Механизм подключения и запуска различных типов зависимостей, реализованный по определенным соглашениям на основе интерфейсных методов и абстрактного класса, описывающего объект «зависимость». Конкретные реализации алгоритмов зависимостей подключаются в виде динамических библиотек. При переходе на определенный узел вычисления, вызывается соответствующая ему библиотека (если доступна в системе), на вход которой передается массив входящих атрибутных связей и выражение, описывающее правило. Результаты используются для вычисления остальных узлов. Механизм позволяет компоновать различные методы вычисления правил, включая аналитические формулы, эвристические правила, в дальнейшем планируется подключение нейросетевых вычислений и других интеллектуальных процедур.
- Реализации конкретных методов вычисления правил. В настоящее время системой поддерживаются два типа процедур – формулы и эвристические правила. Для обработки формул используется математический процессор Java Expression Parser (JEP), а специальное расширение над ним для ограниченной обработки эвристических правил. На следующем этапе разработки системы, для обработки правил планируется использование более совершенных методов, например реализаций алгоритма Rete [11].

Технически, эта подсистема реализована на языке Java, использует для своей работы ряд сторонних открытых библиотек и в данной версии интегрирована с подсистемой моделирования.

Подсистема моделирования представляет собой графический интерфейс, поддерживающий все этапы разработки модели. Интерфейс позволяет описать предмет моделирования в виде набора диаграмм, вводить исходные данные и анализировать результаты.

Начиная новый проект, пользователь может выбрать, какие из уже описанных в общей библиотеке структур (шаблоны подсистем и классов) использовать в качестве оснований для наследования, или непосредственного включения в модель. Все выбранные библиотеки, а также структуры, вносимые в модель пользователем, отражаются на специальной панели, обеспечивающей наглядную навигацию.

Следующим шагом является описание модели подсистем, для чего используется соответствующая панель пользовательского интерфейса, представляющая собой редактор диаграмм (рис. 3). Подсистемы обозначаются блоками, а дугами описываются отношения подчинения или часть-целое. Для каждого блока на диаграмме можно вызвать панель деталей, на которой описывается общая информация о подсистеме, основание декомпозиции, связь с классом объектов и другая информация.

Аналогичным образом создаются модели классов и атрибутов. Выбрав из ресурсов ряд заранее описанных в системе классов, пользователь может расширить их до потребностей предметной области с помощью механизма наследования, при этом все атрибуты и методы суперкласса будут доступны автоматически.

На модели атрибутов присутствуют несколько типов блоков, характеризующих различные способы задания зависимости между атрибутами. При соединении блоков атрибутов и зависимостей с помощью специального инструмента, входящие связи автоматически подставляются в редактор зависимости и становятся доступны для использования в аналитическом выражении.

Описанная последовательность моделей не является обязательной и в любой момент возможно переключение между диаграммами, при этом адекватно отображается изменение общих структурных элементов (например, изменение состава атрибутов класса на диаграмме классов автоматически отражается на остальных моделях). В системе существуют наборы инструментов для графического отображения подсистем, классов, атрибутов, зависимостей и различных связей. Также присутствует возможность «перетаскивания» элементов на диаграмму из панели ресурсов, утилиты для масштабирования, несколько алгоритмов для автоматического упорядочивания и раскладки блоков.

Кроме редактора диаграмм, интерфейс включает в себя ряд служебных панелей для сопровожде-

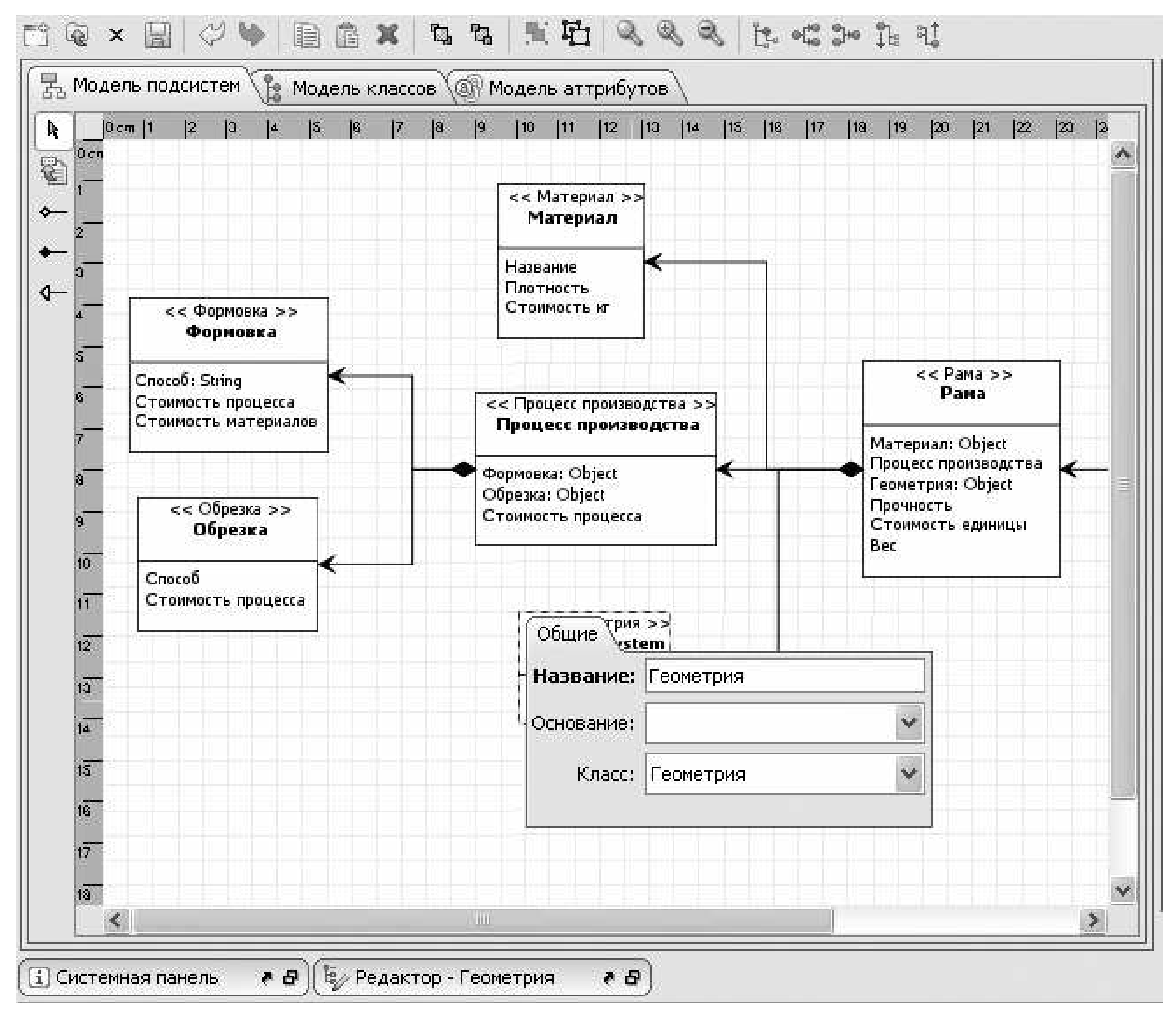


Рис. 3. *Фрагмент пользовательского интерфейса, обеспечивающего редактирование модели подсистем*

ния и анализа модели – ввода данных, получения результатов, построения графиков.

Подсистема моделирования выполнена в виде Java приложения на основе Swing API и некоторых *open-source* компонент, а графическое ядро основано на открытой библиотеке JGraph. В настоящий момент эта подсистема является связующим, интегрирующим звеном в архитектуре приложения, но в дальнейшем планируется перенести механизм вывода и базу знаний на сторону серверного приложения и оставить интерфейс в виде клиентского приложения. Это позволит лучше выполнять ряд требований к современной интеллектуальной системе, приведенных в начале статьи (таких как масштабируемость, переносимость, интеграционный потенциал).

Заключение

Дальнейшим направлением развития инструментального средства является расширение поддерживаемых алгоритмов ветвления на модели и вычисления зависимостей, введение динамики за счет отслеживания состояний модели и интеграции с обновляемыми источниками данных, разработка механизмов сокращения вариантов для перебора при вычислении зависимостей, подключение механизмов учета неопределенности и нечеткости параметров модели.

С технической стороны имеется возможность переноса подсистем вывода и хранения знаний на серверную часть, таким образом, система будет более масштабируемой, с возможностью подключения производительных распределенных вычислительных ресурсов.

Окончательная реализация описанной системы позволит осуществлять быстрое и удобное моделирование проблем в условиях неопределенности, использовать интеллектуальные технологии для поддержки принятия в различных предметных областях.

СПИСОК ЛИТЕРАТУРЫ

1. Попов Э.В. Экспертные системы реального времени [Электронный ресурс] // Открытые системы – 1995. – № 2. – Электрон. дан. – Режим доступа: http://www.osp.ru/text/302/178608/
2. Crossland R., Sims W.J.H., McMahon C.A. An object-oriented modeling framework for representing uncertainty in early variant design. // Research in Engineering Design – 2003. – № 14. – С. 173–183.
3. Landmark Graphics ARIES [Электронный ресурс] – Электрон. дан. – 2006. – Режим доступа: http://www.geographix.com/ps/viewpg.aspx?navigation_id=1273
4. Schlumberger Merak [Электронный ресурс] – Электрон. дан. – 2006. – Режим доступа: http://www.slb.com/content/services/software/valuerisk/index.asp
5. Gensim G2 [Электронный ресурс] – Электрон. дан. – 2006. – Режим доступа: – http://www.gensym.com/?p=what_it_is_g2
6. Thurston D.L., Liu T. Design Evaluation of Multiple Attribute Under Uncertainty // Systems Automation: Research and Applications. – 1991. – V. 1. – № 2. – P. 93–102.
7. Paredis C.J.J., Diaz-Calderon A., Sinha R., Khosla P.K. Composable Models for Simulation-Based Design // Engineering with Computers. – 2001. – № 17. – P. 112–128.
8. Силич М.П. Системная технология: объектно-ориентированный подход. – Томск: Том. гос. ун-т систем управления и радиоэлектроники, 2002. – 224 с.
9. Силич М.П., Стародубцев Г.В. Объектная модель выбора инвестиционных проектов разработки нефтегазовых месторождений. // Автоматизация, телемеханизация и связь в нефтяной промышленности. – 2004. – № 11. – С. 16–21.
10. Хабибулина Н.Ю., Силич М.П. Поиск решений на модели функциональных отношений // Информационные технологии – 2004. – № 9. – С. 27–33.
11. Jess Rete Algorithm [Электронный ресурс] – Электрон. дан. – 2006. – Режим доступа: http://www.jessrules.com/jess/docs/70/rete.html

УДК 651.51